https://johoe.kmu.ac.ir
doi:10.34172/.johoe.2509.1793
JOHOE. 2026;15:2509.1793-S1

In this file, we first provide a detailed description of Overall Prevalence, GEE, REM, and Average Individual Prevalence estimators, and their related confidence intervals. Next, we present a detailed description of the simulation settings. These estimators were evaluated through simulation study.

Prevalence estimators

Overall Prevalence estimator

The population-tooth level prevalence estimator is defined as:

$$\widehat{Prevalence}_{\textbf{Overall Prevalence}} = \frac{total\ number\ of\ teeth\ with\ untreated\ dental\ caries\ in\ sample}{total\ number\ of\ teeth\ \ in\ sample}.$$

This estimator represents a simple proportion. The confidence interval for this estimator, using normal approximation, is given by:

$$\widehat{Prevalence}_{\textbf{Overall Prevalence}} \pm z_{\frac{\alpha}{2}} \sqrt{\frac{\left(\widehat{Prevalence}_{\textbf{Overall Prevalence}}\right) \times \left(1 - \widehat{Prevalence}_{\textbf{Overall Prevalence}}\right)}{total\ number\ of\ teeth\ in\ sample}}.$$

It is important to note that if untreated dental caries for each teeth are considered as a binary variable (Y_{ij} = 0 or 1 with i representing subjects and j representing teeth), and a logistic regression model with only the intercept is used, the estimate obtained from the regression model will equal the simple estimator. The logistic regression without any independent variables has the following form:

$$\ln\left(\frac{P(Y_{ij}=1)}{P(Y_{ij}=0)}\right) = b_0$$

In this way, $P(Y_{ij}=1)$ is equal to prevalence of untreated dental caries, and we can estimate the prevalence of untreated dental caries from the above equation as:

$$\frac{\exp(\widehat{b_0})}{1+\exp(b_0)} \qquad (1)$$

where $\widehat{b_0}$ is maximum likelihood estimator of b_0. To calculate the confidence intervals for the prevalence parameter, one can apply the estimator to the lower and upper bounds of the confidence interval for $\widehat{b_0}$.

R codes to calculated the overall prevalence for the hypothetical example:

```
Ni=c(20,26,7,10,17)
Di=c(2,1,3,4,2)
id=c(1,2,3,4,5)
overall=sum(Di)/sum(Ni)
clUoverall=overall+1.96*sqrt(overall*(1-overall)/sum(Ni))
clLoverall=overall-1.96*sqrt(overall*(1-overall)/sum(Ni))
overall
clUoverall
clLoverall
```

GEE estimator

In the GEE method, the logistic regression equation remains unchanged for obtaining prevalence, but a correlation matrix is used to derive prevalence, and its standard deviation(14). This correlation matrix captures the relationship between untreated dental caries outcomes across each two teeth within the same subject. In statistical software, specifying the correlation structure allows for the estimation of the intercept and correlation coefficients, along with their confidence intervals. As before, prevalence and its confidence interval can be derived using Equation 1.

R codes to calculated the GEE estimator for the hypothetical example:

```
library(geepack)
Ni=c(20,26,7,10,17)
Di=c(2,1,3,4,2)
id=c(1,2,3,4,5)
datalong=data.frame()
n=5
# Loop through each individual to create the binary long dataset
for (i in 1:n) {
  # Create a vector for the current individual
  individual_data <- c(rep(0, Ni[i] - Di[i]), rep(1, Di[i]))

  # Append the individual data to the long_data list
  datalong <- rbind(datalong, data.frame(id = rep(id[i], Ni[i]), decayed = individual_data))
}
```

```
# Fit GEE logistic model
gee_model_logit <- geeglm(
  decayed ~ 1,                    # Prevalence modeled as intercept-only
  id=id,   # Cluster by individual
  family = binomial(link = "logit"), # Logit link
  data = datalong , corstr = "exchangeable"
)
logit_intercept <- coef(gee_model_logit)[1]  # Intercept
prevalence <- exp(logit_intercept) / (1 + exp(logit_intercept))
prevalence
se_logit <- summary(gee_model_logit)$coefficients[, "Std.err"]
z_value <- 1.96  # For 95% CI
lower_logit <- logit_intercept - z_value * se_logit
upper_logit <- logit_intercept + z_value * se_logit
# Convert CI bounds from logit scale to probability
lower_ci <- exp(lower_logit) / (1 + exp(lower_logit))
upper_ci <- exp(upper_logit) / (1 + exp(upper_logit))
prevalence
lower_ci
upper_ci
```

REM estimator

In the logistic regression model, the effects of random prevalence follow an equation similar to logistic regression, with the difference that for each individual in the study, there are two terms in the logistic regression model: one term that is the same for all individuals and another term that varies from one individual to another. In this case, the differences between individuals and the similarities within individuals in the teeth are taken into account in the calculations. The logistic regression model in this way is defined by the following equation:

$$\ln\left(\frac{P(Y_{ij}=1)}{P(Y_{ij}=0)}\right) = b_0 + u_j.$$

In the above equation, the term u_j accounts for individual differences and is assumed to follow a normal distribution with a mean of zero and unknown variance. As before, statistical software calculates the estimate of b_0 along with its confidence interval, which can be used with Equation 1 to estimate prevalence and its confidence interval.

R codes to calculated the REM estimator for the hypothetical example:

```
library(lme4)


Ni=c(20,26,7,10,17)
Di=c(2,1,3,4,2)
id=c(1,2,3,4,5)
datalong=data.frame()


n=5
# Loop through each individual to create the binary long dataset
for (i in 1:n) {
  # Create a vector for the current individual
  individual_data <- c(rep(0, Ni[i] - Di[i]), rep(1, Di[i]))

  # Append the individual data to the long_data list
  datalong <- rbind(datalong, data.frame(id = rep(id[i], Ni[i]), decayed = individual_data))
}

# Fit REM logistic model

REM_model_logit <- glmer(decayed ~ 1 + (1 | id),
              family = binomial(link = "logit"),
              data = datalong )
logit_intercept <- summary(REM_model_logit)$coefficients[, "Estimate"]
prevalence <- exp(logit_intercept) / (1 + exp(logit_intercept))

se_logit <- summary(REM_model_logit)$coefficients[, "Std.err"]
```

```
z_value <- 1.96  # For 95% CI
lower_logit <- logit_intercept - z_value * se_logit
upper_logit <- logit_intercept + z_value * se_logit

# Convert CI bounds from logit scale to probability
lower_ci <- exp(lower_logit) / (1 + exp(lower_logit))
upper_ci <- exp(upper_logit) / (1 + exp(upper_logit))
prevalence
lower_ci
upper_ci
```

Average Individual Prevalence estimator

From the perspective of cluster sampling, individuals in the study are considered as clusters, sampled through simple random sampling. After identifying the clusters, all members of each cluster (the teeth of each individual) are evaluated for untreated decay. Thus, to calculate prevalence, the prevalence is first computed within each cluster, and then the obtained prevalences are combined based on the cluster weights. Since the clusters are selected through simple random sampling, the weights of the clusters are equal and correspond to 1 divided by the number of individuals in the study. Therefore, the Average Individual Prevalence estimator is obtained as follows.

$$\widehat{Prevalence}_{\textbf{Average Individual Prevalence}} = \frac{\sum \frac{number\ of\ untreated\ dental\ caries\ for\ i^{th}\ person\ in\ sample}{total\ number\ of\ teeth\ \ for\ i^{th}\ person\ in\ sample}}{total\ number\ of\ participants}$$

The estimation of variance and confidence intervals in this method is extensively discussed in sampling textbooks. However, a very simple way to obtain the confidence interval for the cluster estimate is to use the bootstrap method. Statistical software easily provides confidence interval estimates using the bootstrap method.

R codes to calculated the Average Individual Prevalence estimator for the hypothetical example:

```
statistic_function <- function(data, indices) {
  # Subset the data
  sample_data <- data[indices, ]
```

```
  # Calculate Di/Ni for each row
  ratios <- sample_data$Di / sample_data$Ni

  # Return the mean of the ratios
  return(mean(ratios))
}
bootstrap_results <- boot(data = datashort, statistic = statistic_function, R = 1000)
bootstrap_ci<- boot.ci(bootstrap_results, type = "perc")



individual=unname(unlist(bootstrap_results[1]))

a=unlist(bootstrap_ci)
clUindividual=a$percent5
clLindividual=a$percent4
individual
clUindividual
clLindividual
```